The ISC Int'l Journal of Information Security

ISeCure

January 2010, Volume 2, Number 1 (pp. 33–46)

http://www.isecure-journal.org

A Hybridization of Evolutionary Fuzzy Systems and Ant Colony Optimization for Intrusion Detection

Mohammad Saniee Abadeh [a,*], Jafar Habibi [a]

[a] *Department of Computer Engineering, Sharif University of Technology, Tehran, Iran*

ARTICLE INFO.

Article history:
Received: 21 April 2009
Revised: 9 December 2009
Accepted: 15 December 2009
Published Online: 26 January 2010

Keywords:
Intrusion Detection System, Evolutionary Fuzzy System, Ant Colony Optimization, Fuzzy Rule Extraction

ABSTRACT

A hybrid approach for intrusion detection in computer networks is presented in this paper. The proposed approach combines an evolutionary-based fuzzy system with an Ant Colony Optimization procedure to generate high-quality fuzzy-classification rules. We applied our hybrid learning approach to network security and validated it using the DARPA KDD-Cup99 benchmark data set. The results indicate that in comparison to several traditional and new techniques, the proposed hybrid approach achieves better classification accuracies. The compared classification approaches are C4.5, Naïve Bayes, k-NN, SVM, Ripper, PNrule and MOGF-IDS. Moreover the improvement on classification accuracy has been obtained for most of the classes of the intrusion detection classification problem. In addition, the results indicate that the proposed hybrid system's total classification accuracy is 94.33% and its classification cost is 0.1675. Therefore, the resultant fuzzy classification rules can be used to produce a reliable intrusion detection system.

1 Introduction

Intrusion Detection Systems (IDS) act as the "second line of defense" placed inside a protected network, looking for known or potential threats in network traffic and/or audit data recorded by hosts [1].

The problem of intrusion detection has been studied extensively in computer security [2–5], and has received a lot of attention in machine learning and data mining [6–8].

Intrusion detection is classified into two types: misuse intrusion detection and anomaly intrusion detection.

The misuse intrusion detection is a rule-based approach that uses stored signatures of known intrusion instances to detect an attack [9–11]. This approach is highly successful in detecting occurrences of previously known attacks. However, it fails to detect new attack types and variants of known attacks whose signatures are not stored. When new attacks occur, the signature database has to be manually modified for future use. In anomaly intrusion detection, usually a profile for normal behavior is first established [12–14]. Then deviants from the normal profile are considered as anomalies. In some cases, these anomalies may be just normal operations that are exhibiting some behaviors adherent to unseen mode of operation. In such cases, the anomalies may be showing false positives. That is, classifying a normal behavior as abnormal, and hence as possible attack instances.

The above discussion points out that the tradeoff between the ability to detect new attacks and the

* Corresponding author. He is currently with Faculty of Electrical and Computer Engineering, Tarbiat Modares University, Tehran, Iran.

Email addresses: saniee@modares.ac.ir (M. Saniee Abadeh), jhabibi@sharif.edu (J. Habibi)

ISSN: 2008-2045

ability to generate a low rate of false alarms is the key point to develop an effective IDS. In this paper, we exploit a new hybrid evolutionary fuzzy system to develop an IDS based on misuse detection. The goal of our algorithm is to find high quality fuzzy if-then rules to predict the class of input patterns correctly.

The objective of this paper is to extract fuzzy classification rules for misuse intrusion detection in computer networks. The presented learning system consists of two stages. In the first stage, an iterative rule-learning algorithm is applied to the training data to generate a primary set of fuzzy rules. In this stage, the fuzzy rule base is generated in an incremental fashion, in that the evolutionary algorithm optimizes one fuzzy classifier rule at a time. The second stage of the algorithm employs an ant colony optimization procedure to enhance the quality of the primary fuzzy rule set from the previous stage. As our proposed classification algorithm is a learning system that hybridizes an Evolutionary Fuzzy System with an Ant Colony Optimization procedure, for simplicity, through the rest of paper we call it EFS-ACO.

The proposed hybrid evolutionary fuzzy system was evaluated using the KDD-Cup99 benchmark dataset—which contains information on computer networks, during normal and intrusive behaviors. This dataset is available at the University of California, Irvine web site [15].

The rest of this paper is organized as follows: Related work on intrusion detection is introduced in Section 2. Fuzzy rule base for pattern classification is presented in Section 3. The first stage of the proposed learning algorithm, which is an evolutionary fuzzy system, is discussed in Section 4. Section 5 describes the presented Ant Colony Optimization procedure as the second stage of EFS-ACO in detail. Experimental results are reported in Section 6, and in Section 7 we will discuss about the main idea behind EFS-ACO. Finally, Section 8 concludes the paper.

2 Related Work

There are several approaches for solving intrusion detection problems. Lee [16] built an intrusion-detection model using association-rule and frequent-episode techniques on system audit data.

Neural networks have been extensively used to detect both misuse and anomalous patterns [17–21]. An n-layer network is constructed and abstract commands are defined in terms of sequences of information units, the input to the neural network in the training data.

Each command, together with some predefined commands, is used to predict upcoming command expected from the user. After training, the system has the profile of the user. At the testing step, the anomaly is said to occur when the user deviates from the expected behavior [22, 23]. Short sequences of system calls carry out the prediction process. In this system, Hamming distance comparison with a threshold is used to discriminate the normal sequence from the abnormal sequence [24, 25]. Some recent researches have utilized Artificial Immune Systems to detect intrusive behaviors in a computer network [26–28].

Some other applied techniques on intrusion detection problem are Genetic Fuzzy Rule-Based Systems [11, 12, 29], Bayesian parameter estimation [9] and clustering [30–33].

This paper hybridizes an iterative evolutionary fuzzy system with an ant colony optimization procedure to extract fuzzy classification rules for intrusion detection problem. Note that this hybridization is the main contribution of this paper. In other words, the use of hybrid nature-inspired heuristics to extract fuzzy classification rules has not been investigated in previous works.

3 Fuzzy Rule Base for Pattern Classification

Let us assume that our pattern classification problem is a c-class problem in the n-dimensional pattern space with continuous attributes. We also assume that M real vectors $x_p = (x_{p1}, x_{p2}, \ldots, x_{pn})$, $p = 1, 2, \ldots, M$, are given as training patterns from the c classes ($c \ll M$).

Because the pattern space is $[0, 1]^n$, attribute values of each pattern are $x_{pi} \in [0, 1]$ for $p = 1, 2, \ldots, M$ and $i = 1, 2, \ldots, n$. In computer simulations of this paper, we normalize all attribute values of each data set into the unit interval $[0, 1]$. This process is done since we want to apply same membership functions to all of the attributes in the intrusion detection classification problem. The above mentioned normalization will make our final classification system more comprehensive and interpretable.

In the presented fuzzy classifier system, we use fuzzy if-then rules of the following form.

Rule R_j: If x_1 is A_{j1}, and x_2 is A_{j2}, ..., and x_n is A_{jn}, then Class C_j with $CF = CF_j$.

where R_j is the label of the jth fuzzy if-then rule, A_{j1}, ... , A_{jn} are antecedent fuzzy sets on the unit interval $[0, 1]$, C_j is the consequent class (i.e., one of the given c classes), and CF_j is the grade of certainty of the fuzzy if-then rule R_j. In computer simulations, we use a typical set of linguistic values in Figure 1 as

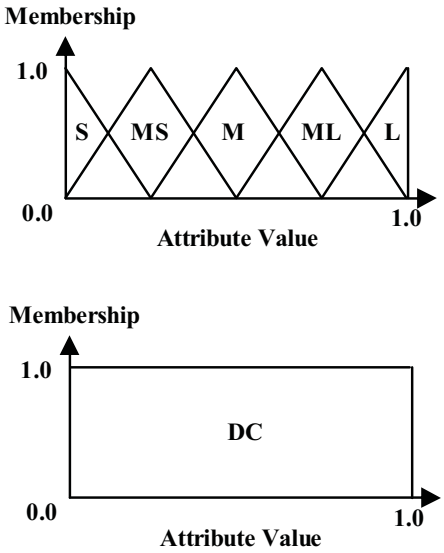


Figure 1. The antecedent fuzzy sets used in this paper. 1: Small, 2: medium small, 3: medium, 4: medium large, 5: large, and 0: don't care.

antecedent fuzzy sets. The membership function of each linguistic value in Figure 1 is specified by homogeneously partitioning the domain of each attribute into symmetric triangular fuzzy sets. We use such a simple specification in computer simulations to show the high performance of our fuzzy classifier system, even if the membership function of each antecedent fuzzy set is not tailored. However, we can use any tailored membership functions in our fuzzy classifier system for a particular pattern classification problem.

According to Figure 1, the total number of fuzzy if-then rules is 6^n in the case of the n-dimensional pattern classification problem. It is impossible to use all the 6^nfuzzy if-then rules in a single fuzzy rule base when the number of attributes (i.e.n) is large (e.g., intrusion detection problem which $n = 41$). Our proposed hybrid system searches for a relatively small number of fuzzy if-then rules (e.g., 50 rules) with high classification ability.

The first stage of EFS-ACO that generates and evolves a primary fuzzy rule set, performs its searching process for each of the c classifiers independently. (c denotes the number of classes in the classification problem). Each classifier contains a subset of rules with the same class labels. The proposed algorithm focuses on learning of each class to improve the total accuracy of the main classifier. Therefore, the proposed hybrid system is repeated for each class of the classification problem separately [34].

The second stage of EFS-ACO attempts to improve the total accuracy of the primary fuzzy rule set. This primary rule set is the result of searching process from the first stage. In the second stage of EFS-ACO, an ant colony optimization procedure is used to search the neighborhood of the each fuzzy rule in the primary fuzzy rule set and improves that rule according to the total classification rate of the primary rule set.

Since the learning process of each class in the first stage of the proposed hybrid system is independent from the other classes, therefore the class label of each fuzzy rule is determined according to its corresponding learning process. In other words, we do not need to compute the class label of each fuzzy if-then rule in the first stage, because every rule in the population would have the class label that the learning process considers more fitness value to it.

The EFS-ACO applies the following steps to calculate the certainty grade of each fuzzy if-then rule [34]:

Step 1: Calculate the compatibility of each training pattern $x_p = (x_{p1}, x_{p2}, \ldots, x_{pn})$ with the fuzzy if-then rule R_j by the following product operation:

$$\mu_j(x_p) = \mu_{j1}(x_{p1}) \times \cdots \times \mu_{jn}(x_{pn}), \qquad p = 1, 2, \ldots, m \tag{1}$$

where $\mu_{ji}(x_{pi})$ is the membership function of i^{th} attribute of p^{th} pattern and M denotes the total number of patterns.

Step 2: For each class, calculate the relative sum of the compatibility grades of the training patterns with the fuzzy if-then rule R_j:

$$\beta_{\text{Class } h}(R_j) = \sum_{x_p \in \text{Class } h} \frac{\mu_j}{N_{\text{Class } h}}, \qquad h = 1, 2, \ldots, m \tag{2}$$

where $\beta_{\text{Class } h}(R_j)$ is the sum of the compatibility grades of the training patterns in *Class h* with the fuzzy if-then rule R_j and $N_{\text{Class } h}$ is the number of training patterns which their corresponding class is *Class h*.

Step 3: The grade of certainty CF_j is determined as follows:

$$CF_j = \frac{\left(\beta_{\text{Class } \hat{h}_j}(R_j) - \bar{\beta}\right)}{\sum_{h=1}^{c} \beta_{\text{Class } h}(R_j)} \tag{3}$$

where

$$\bar{\beta} = \frac{\sum_{h \neq \hat{h}_j} \beta_{\text{Class } h}(R_j)}{c - 1} \tag{4}$$

By the proposed heuristic procedure, we can specify the certainty grade for any combination of antecedent

fuzzy sets. Such a combination is generated by the proposed hybrid system will be explained in the next sections.

The task of our fuzzy classifier system is to generate combinations of antecedent fuzzy sets for generating a rule set S with high classification ability. When a rule set S is given, an input pattern $x_p = (x_{p1}, x_{p2}, ..., x_{pn})$ is classified by a single winner rule R_{j^*} in S, which is determined as follows [35]:

$$\mu_{j^*}(x_p) \cdot CF_{j^*} = \max\{\mu_j(x_p) \cdot CF_j \mid R_i \in S\} \quad (5)$$

That is, the winner rule has the maximum product of the compatibility and the certainty grade CF_j.

The method of coding fuzzy if-then rules, which is used in this paper, is the same as the method that we employed in [10]. Each fuzzy if-then rule is coded as a string. The following symbols are used for denoting the five linguistic values: (Figure 1)

0: don't care (DC), 1: small (S), 2: medium small (MS), 3: medium (M), 4: medium large (ML), 5: large (L).

As it is mentioned earlier, the proposed learning algorithm combines an evolutionary fuzzy system with an ant colony optimization procedure to generate its final fuzzy rule set. We will discuss about each of these two stages in the following sections.

4 Stage I: Evolutionary Fuzzy System

The first stage of EFS-ACO is an evolutionary fuzzy system that learns fuzzy if-then rules in an incremental fashion, in that the evolutionary algorithm optimizes one fuzzy classifier rule at a time. The learning mechanism decreases the weight of those training instances that are correctly classified by the new rule. Therefore, the next rule generation cycle focuses on fuzzy rules that account for the currently uncovered or misclassified instances. At each iteration the fuzzy rule that can classifies the current distribution of training samples better than other rules of the population is selected out to be included in the final classification fuzzy rule base. The idea behind using the boosting mechanism is to aggregate multiple hypotheses generated by the same learning algorithm invoked over different distributions of the training data into a single composite classifier.

In the above learning framework, we have used the fitness function, which is computed according to equations (6) to (8).

$$f_p = \frac{\sum_{k|c^k=c_i} \left(w^k \cdot \mu_{R_i}(x^k)\right)}{\sum_{k|c^k=c_i} w^k} \quad (6)$$

$$f_N = \frac{\sum_{k|c^k\neq c_i} \left(w^k \cdot \mu_{R_i}(x^k)\right)}{\sum_{k|c^k\neq c_i} w^k} \quad (7)$$

$$\text{fitness}(R_j) = w_p f_p - w_N f_N \quad (8)$$

where,

f_p: rate of positive training instances covered by the rule R_i (correct classification).

f_N: rate of negative training instances covered by the rule R_i (misclassification).

w^k: a weight which reflects the frequency of the instance x^k in the training set.

w_p: the weight of positive classification.

w_N: the weight of negative classification (misclassification).

Outline of the proposed iterative evolutionary fuzzy system is presented as follows:

Step 1: Generate an initial population of fuzzy if-then rules based on weight of training samples. (Initialization)

Step 2: Generate new fuzzy if-then rules by genetic operations. (Generation)

Step 3: Replace a part of the current population with the newly generated rules. (Replacement)

Step 4: Terminate the inner cycle of the algorithm if a stopping condition is satisfied, otherwise return to Step 2. (Inner Cycle Termination Test)

Step 5: Terminate the outer cycle of the algorithm if a stopping condition is satisfied, otherwise go to the next step (Outer Cycle Termination Test)

Step 6: Adjust the new weight of each training sample that covers by the new added fuzzy rule. Go to Step 1 (Weight Adjustment).

Each step of EFS-ACO is described as follows:

Step 1: Let us denote the number of fuzzy if-then rules in the population by N_{pop}. To produce an initial population, N_{pop} fuzzy if-then rules are generated according to a random pattern in the train dataset [36]. As it was mentioned in the previous section, the proposed evolutionary fuzzy system is considered for each of the classes of the classification problem separately. Therefore, the mentioned random pattern is extracted according to the patterns of the training dataset, which their consequent class is the same as the class that the algorithm works on. Note that the probability for each training pattern to be chosen in

this step is proportional to its current weight. This means that the algorithm considers a greater probability for those patterns that have not been learned in previous iterations. Next, for this random pattern, we determine the most compatible combination of antecedent fuzzy sets using only the five linguistic values (Figure 1). The compatibility of antecedent fuzzy sets with the random pattern is measured by equation (1). After generating each fuzzy if-then rule, the certainty grade of this rule is determined according to the heuristic method, explained in the previous section. After generation of N_{pop} fuzzy if-then rules, the fitness value of each rule is evaluated by classifying all the given training patterns using the set of fuzzy if-then rules in the current population. Each fuzzy if-then rule is evaluated according to the fitness function, which is presented in equation (8).

Step 2: A pair of fuzzy if-then rules is selected from the current population to generate new fuzzy if-then rules for the next population. Each fuzzy if-then rule in the current population is selected using the tournament selection strategy. This procedure is iterated until a pre-specified number of pairs of fuzzy if-then rules are selected. A crossover operation is then applied to a selected random pair of fuzzy if-then rules with a pre-specified crossover probability. Note that the selected individuals for crossover operation should be different. In computer simulations of this paper, we have used the uniform crossover. With a pre-specified mutation probability, each antecedent fuzzy set of fuzzy if-then rules is randomly replaced with a different antecedent fuzzy set after the crossover operation. Note that the probability of changing to don't care value is more than the other five linguistic values. We call this probability P_{DC}. After performing selection, crossover and mutation steps, the fitness value of each of the generated individuals is evaluated according to equation (8).

Step 3: A pre-specified number of fuzzy if-then rules in the current population are replaced with the newly generated rules. In our fuzzy classifier system, P_R percent of the worst rules with the smallest fitness values are removed from the current population and $(100 - P_R)$ percent of the newly generated fuzzy if-then rules are added. (P_R is the replacement percentage) After performing the mentioned replacement procedure, the fitness value of each of the individuals is evaluated according to equation (8).

Step 4: We can use any stopping condition for terminating the inner cycle of the IRL-based fuzzy rule learning algorithm. In computer simulations of this paper, we used the total number of generations as a stopping condition.

Step 5: After termination of the inner cycle of EFS-ACO, the algorithm adds the best fuzzy rule of the evolved population to the final classification rules list and checks if this added fuzzy rule is capable of improving the classification rate of final classification system. If the classification rate is not improved the algorithm removes the added fuzzy rule from the final classification rules list and terminates. Otherwise, it goes to the next step.

Step 6: At each iteration of the main evolutionary process, rule R_t with the best fitness value is inserted into the primary fuzzy rule base. After each rule extraction process, instances that are misclassified will end up having the same weight. The weight of those instances that are classified correctly will be became zero. Note that initially $w^k = 1$. After this step, the algorithm jumps to step 1.

5 Stage II: ACO Meta-Heuristic

ACO is a stochastic approach that has been proposed to solve different hard combinatorial optimization problems such as traveling salesman problems [37–39] graph coloring problems [40], quadratic assignment problems [41, 42] and vehicle routing problems [43, 44]. The main idea of ACO is to model the problem as the search for a minimum cost path in a graph. Artificial ants walk through this graph, looking for good paths. Each ant has a rather simple behavior so that it will typically only find rather poor-quality paths on its own. Better paths are found as the emergent result of the global cooperation among ants in the colony. This global cooperation is performed using pheromone trails, which ants deposit whenever they travel, as a form of indirect communication.

The behavior of artificial ants is inspired from real ants [45]: they lay pheromone trails on the graph edges and choose their path with respect to probabilities that depend on pheromone trails and these pheromone trails progressively decrease by evaporation. In addition, artificial ants have some extra features that do not find their counterpart in real ants. In particular, they live in a discrete world (a graph) and their moves consist of transitions from nodes to nodes. In addition, they are usually associated with data structures that contain the memory of their previous actions. In most cases, pheromone trails are updated only after having constructed a complete path and not during the walk, and the amount of pheromone deposited is usually a function of the quality of the path. Finally, the probability for an artificial ant to choose an edge often depends not only on pheromones, but also on some problem-specific local heuristics.

In order to improve the classification rate of the resulted fuzzy rule set from the first stage of EFS-

ACO, a secondary stage is considered. In this stage, an ant colony optimization procedure is applied on the primary fuzzy rule set from the first stage of EFS-ACO. This secondary stage of EFS-ACO acts as a local search procedure in that the ACO improves one fuzzy rule from the primary rule set at a time. The so-called improvement is done by modifying at most M antecedents of the candidate fuzzy rule from the primary rule set. M denotes maximum allowed possible modifications to the candidate fuzzy rule. The mentioned candidate rule from the primary rule set is selected according to a selection strategy. Outline of this selection strategy is as follows:

Step 1: Sort the rules of primary fuzzy rule set according to their class and their sequence of generation in the first stage of EFS-ACO. In other words, the rules of each class should be followed with one another according to their generation sequence.

Step 2: $t = 1$.

Step 3: Perform the ACO-based local search procedure on R_t. If after updating R_t, the resulted rule set has a higher classification rate than the primary rule set then accept the new R_t and update the primary rule set. Otherwise, reject the new R_t.

Step 4: If the new R_t is accepted then:

Step 4.1: $b = t$.

Step 4.2: Go to step 3 (Since the local search procedure has been successful on R_t, therefore the selection mechanism repeats the local search procedure for R_t).

Step 5: $t = t + 1$. If $t >$ primary rule set size then $t = 1$.

Step 6: If $|b - t| = S$ then terminate the algorithm. S denotes the size of primary fuzzy rule set (number of rules in it).

Step 7: Go to Step 3.

The operation of ACO-based local search procedure, that searches the neighborhood of the current candidate rule and finds a proper modification for it, is as follows:

The ACO-based local searcher employs a population of ants to perform its local search process. This population will searches the neighborhood of the candidate rule and improves it according to the best-discovered modification. Since we are using the ACO as a local search procedure, the representation of ants pheromone trails (paths) must be of a form that can show an instruction of how to change the candidate rule to improve the total accuracy (classification rate) of the primary rule set. Hence, each path is a string of characters that shows those parts of the candidate rule that should be modified. Note that the above discussion implies that the evaluation of each path is done according to the amount of improvement of the primary rule set classification rate.

The detail of ACO-based local search procedure is as follows:

Step 1: Initiate the pheromone trails and parameters (e.g. iteration = 0). Note that in this initiation the amount of initial pheromone is $\tau_0 = 0.004$. This value for the initial pheromone is considered according to (9) [46]:

$$\tau_{ij}(t=0) = \frac{1}{\sum_{i=1}^{a} b_i} \tag{9}$$

where a is the total number of attributes (e.g. $a = 41$) and b_i is the number of possible values that can be taken on by attribute A_i (e.g. $b_{ij} = 6$).

Step 2: Each ant in the ACO-based local search procedure generates a modification sequence according to M (maximum allowed possible modifications to the candidate fuzzy rule). The desirability of each ant for changing the i^{th} antecedent of the candidate rule R_C to A_j is given by (10).

$$p_a(R_C, i, A_j) = \frac{[\tau(R_C, i, A_j)]\,[\eta(R_C, i, A_j)]}{\sum_{u=0}^{5} [\tau(R_C, i, A_u)]\,[\eta(R_C, i, A_u)]} \tag{10}$$

In this equation τ is the pheromone and η is a heuristic probability, which is 0.5 for $A_j = DC$ and 0.1 for each of the other linguistic values in Figure 1. Therefore the tendency of ants to generate "don't care" is high specifically at the first iterations of the local search procedure.

Step 3: When each ant in the colony generates a modification, the fitness of each modification is calculated and the best so found modification in the local search procedure is updated if a better modification is found. The fitness of each modification is calculated according to the improvement of the classification rate of the original rule set. Note that if the classification rate of the original rule set decreases then the fitness value would be negative. In this situation, the algorithm assigns a zero value for the fitness.

Step 4: Increasing the amount of pheromone of used antecedent values according to (11).

$$\tau(R_C, i, A_j) \leftarrow \tau(R_C, i, A_j) \cdot [1 + \cdot f(R_C, i, A_j)] \tag{11}$$

where $f(R_C, i, A_j)$ denotes the fitness of modification

that changes i^{th} antecedent part of R_C to A_j.

Step 5: Decreasing the amount of pheromones of unused antecedent values to simulate pheromone evaporation in real ant colonies. In the ACO-based local search procedure, pheromone evaporation is implemented in a somewhat indirect way. More precisely, the effect of pheromone evaporation for unused terms is achieved by normalizing the value of each pheromone $\tau(R_C, i, A_j)$. This normalization is performed by dividing the value of each $\tau(R_C, i, A_j)$ by the summation of all $\tau(R_C, i, A_j)$, $\forall i, j$.

Step 6: If the maximum number of iterations is reached then return the best so found modification and terminate the local search procedure.

Step 7: iteration = iteration +1. Go to Step 2.

In the next section, we will investigate the performance of proposed hybrid system according to intrusion detection classification problem.

6 Experimental Results

Experiments were carried out on a subset of the database created by DARPA in the framework of the 1998 Intrusion Detection Evaluation Program [47]. We used the subset that was pre-processed by the Columbia University and distributed as part of the UCI KDD Archive [15]. The available database is made up of a large number of network connections related to normal and malicious traffic. Each connection is represented with a 41-dimensional feature vector. Connections are also labeled as belonging to one out of five classes. One of these classes is the normal class and the rest indicates four different intrusion classes: U2R, R2L, DOS and PRB. These intrusion classes are a classification of 22 different types of attacks in a computer network. Table 1 presents the classification of different types of attacks in the four different intrusion classes and the distribution of each class in the 10% of the KDD-Cup 99 data set.

As shown in Table 1, the number of records in the 10% data set is very large (494021). Also, the proportion of samples per class is not uniform, for example from class U2R the number of samples in the training dataset is 52 while from class DOS the number of samples is 391458. According to this fact we have used a subset of this large dataset as our train dataset, hence the training data set contains 752 randomly generated samples. The KDD-Cup99 independent test data (311029 records) with different class probability distribution and new attacks is used for test set and evaluation of the EFS-ACO with other algorithms. The distribution of different classes in the train and test datasets is presented in Table 2.

Table 1. Classes in the 10% of the KDD-Cup 99 Dataset

Class	Sub-Classes	Samples
NORMAL		97278 (19.6911%)
U2R	buffer_overflow, loadmodule, multihop, perl, rootkit	52 (0.0105%)
R2L	ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster	1126 (0.2279%)
DOS	back, land, neptune, pod, smurf, teardrop	391458 (79.2391%)
PRB	ipsweep, nmap, portsweep, satan	4107 (0.8313%)

Table 2. Distribution of Different Classes in the Train and Test Datasets

Class	Train	Test
NORMAL	100	60593
U2R	52	228
R2L	200	16189
DOS	300	229853
PRB	100	4166

We normalized the train and test data sets, where each numerical value in the data set is normalized between 0.0 and 1.0 according to the following equation:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{12}$$

Hence, 41 numeric features are constructed and normalized to the interval $[0, 1]$. They are given in Table 3. This section consists of two subsections. First we present some experiments of applying proposed EFS-ACO to the intrusion detection classification problem. In the next subsection, we compare the performance of our classifier to several traditional and new classification systems for the intrusion detection classification problem.

6.1 Experiments using EFS-ACO

This subsection presents the experiments of applying EFS-ACO on the intrusion detection dataset. Table 4 shows parameter specification that we have used in our computer simulations for EFS-ACO.

Table 5 is the confusion matrix of EFS-ACO. The top-left entry of Table 5 shows that 57990 instances of the actual NORMAL test set were detected to be normal; the last column indicates that 96% of the actual NORMAL samples were detected correctly. In

Table 3. Feature Set of the Preprocessed KDD-Cup99 Dataset. *Feature Type:* 'S' For Symbolic Feature and 'C' for Continuous.

#	Feature Name	Type	#	Feature Name	Type
1	Duration	C	22	is_guest_login.	S
2	protocol_type	S	23	Count	C
3	service	S	24	srv_count	C
4	flag	S	25	serror_rate	C
5	src_bytes	C	26	srv_serror_rate.	C
6	dst_bytes	C	27	rerror_rate	C
7	land	C	28	srv_rerror_rate.	C
8	wrong_fragment	C	29	same_srv_rate.	C
9	urgent	C	30	diff_srv_rate	C
10	hot	C	31	srv_diff_host_rate	C
11	num_failed_logins	C	32	dst_host_count.	C
12	logged_in	S	33	dst_host_srv_count	C
13	num_compromised	C	34	dst_host_same_srv_rate	C
14	root_shell	C	35	dst_host_diff_srv_rate	C
15	su_attempted	C	36	dst_host_same_src_port_rate	C
16	num_root.	C	37	dst_host_srv_diff_host_rate	C
17	num_file_creations	C	38	dst_host_serror_rate	C
18	num_shells	C	39	dst_host_srv_serror_rate	C
19	num_access_files	C	40	dst_host_rerror_rate	C
20	num_outbound_cmds	C	41	dst_host_srv_rerror_rate	C
21	Is_host_login	C			

Table 4. Parameters Specification in Computer Simulations

Parameter	Value
population size (N_{pop})	100
don't care replacement rate (P_{DC})	0.5
crossover probability (P_X)	0.9
mutation probability (P_M)	0.4
fitness positive weight (W_P)	0.1
fitness negative weight (W_N)	0.9
number of local search iterations	10
number of ants (N_{ant})	50
maximum allowed modifications to the primary rule	5
replacement percentage (P_R)	10
maximum generations of the main algorithm	50

the same way, for R2L, 5414 instances of the actual 'attack' test set were correctly detected. The last column indicates that 33.45% of the actual R2L samples were detected correctly. The bottom row shows that 83.46% of the test set said to be NORMAL indeed were NORMAL and 80.42% of the test set classified,

Table 5. Confusion Matrix of EFS-ACO.

	Detected Class					
Real Class	NORMAL	U2R	R2L	DOS	PRB	%
NORMAL	57990	1385	448	110	473	96
U2R	5	166	32	0	25	72.8
R2L	10319	249	5414	121	81	33.45
DOS	904	247	823	226562	702	98.83
PRB	257	32	15	216	3263	86.25
%	83.46	7.98	80.42	99.8	71.8	**94.33**

as R2L indeed belongs to R2L. The bottom-right entry of the Table 5 shows that 94.33% of the all patterns in the test dataset are correctly classified.

Table 6 represents the cost matrix that defines the cost for each type of misclassification [48].

We aim at minimizing that cost function. Given the confusion and cost matrixes, we calculated the cost of EFS-ACO for the intrusion detection classification problem as shown in Table 7.

Table 6. Cost Matrix Used to Evaluate the Confusion Matrix Related to EFS-ACO [31].

	Detected Class				
Real Class	NORMAL	U2R	R2L	DOS	PRB
NORMAL	0	2	2	2	1
U2R	3	0	2	2	2
R2L	4	2	0	2	2
DOS	2	2	2	0	1
PRB	1	2	2	2	0

Table 7. Cost-Based Scoring of the EFS-ACO.

	Detected Class					
Real Class	NORMAL	U2R	R2L	DOS	PRB	
NORMAL	0	2770	896	220	473	
U2R	15	0	64	0	50	
R2L	41276	498	0	242	162	
DOS	1808	494	1646	0	702	
PRB	257	64	30	432	0	
						0.1675

The bottom-right entry of the Table 7 shows that the classification cost of our algorithm is 0.1675.

The classification rate progress for the whole classification system during several iterations of EFS-ACO is investigated in Figure 2. According to this figure, we can comprehend that our proposed learning algorithm is capable of evolving fuzzy if-then rules that cooperate and compete with one another efficiently. Moreover, the ACO-based local search procedure, which is the secondary stage of EFS-ACO, continues the learning process of the first stage of the main hybrid system efficiently.

6.2 Comparison of EFS-ACO with Other Algorithms

This subsection will compare the performance of EFS-ACO on the intrusion detection classification problem with several traditional and new classification systems. Classification performance of EFS-ACO is measured and compared with that of different baseline classifiers including pruning C4.5, Nave Bayes (NB), k-Nearest Neighbor (k-NN), Support Vector Machine (SVM), Ripper, PNrule and Multi-Objective Genetic Fuzzy Intrusion Detection System (MOGF-IDS) [29]. In k-NN parameter k is set to 5, and the SVM is trained using the well-known fast sequential minimal optimization method with a polynomial kernel. The results are also compared with the winner of KDD-Cup99 contest [48], RIPPER and an improved PNrule [49] classifier recently proposed in the literature. Table 8 shows the results of Recall, Precision, and F-measure of different classifier for each class and Table 9 represents the accuracies and classification costs of the above algorithms.

Table 9. Accuracy and Classification Cost of Different Classifiers.

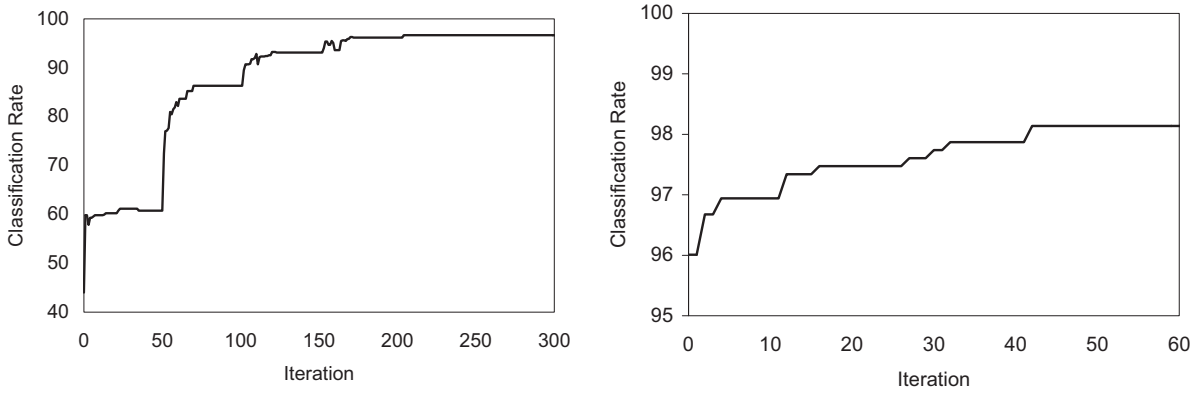
Algorithm	Accuracy	Classification Cost
C4.5	92.04	0.2480
NB	76.45	0.4965
5-NN	91.83	0.2458
SVM	92.54	0.2457
MOGF-IDS [29]	**92.77**	**0.2317**
Winner Entry [48]	92.71	0.2331
EFS-ACO	**94.33**	**0.1675**

Table 8 shows that for the normal traffic recognition EFS-ACO obtains better Precision and F-measure rates than other classifiers. For the class of U2R attacks EFS-ACO outperforms other classifiers in term of Recall. Considering the major class, which is the DOS attacks, the proposed algorithm can achieve the highest Recall and F-measure rates. Regarding the R2L class that is the most important attack according to Table 6, EFS-ACO obtains acceptable Recall and F-measure rates.

According to the Table 9, the EFS-ACO obtains the highest accuracy and the lowest classification cost among other classifiers. This is because of that the EFS-ACO explores the high-dimensional search space of the intrusion detection classification problem more efficiently than other algorithms.

7 Discussion

In this section we discuss about the main idea behind our new hybrid classification system. As it was mentioned in the previous section, EFS-ACO outperforms several well-known classification systems. The main reason to this fact is that we have separated the two important problems in the intrusion detection classification problem which are: finding accurate classification rules (competition) and tuning the learned rules for better classification accuracy for the whole rule set (cooperation). Since our main idea that led us to EFS-ACO depends strongly on our previous works on the use of evolutionary fuzzy systems in the intrusion



(a) (b)

Figure 2. Classification rate progress for the whole classification system during several iterations of EFS-ACO. (a) Classification rate progress during the evolutionary fuzzy rule learning stage (first stage of EFS-ACO). (b) Classification rate progress during the ACO-based local search procedure (second stage of EFS-ACO).

Table 8. Recall, Precision, and F-Measure for Different Classifiers. The Best Values are Bold-Underlined, The Second Bests are Bold, and the Third Bests are Underlined.

Class	Algorithm	C4.5	NB	5-NN	SVM	MOGF-IDS [29]	Winner Entry [48]	Ripper [49]	PNrule [49]	EFS-ACO
NORMAL	Recall	**98.38**	55.47	95.89	97.99	98.36	**99.50**	N/A	N/A	96
	Precision	**74.75**	43.33	74.15	73.42	74.74	74.61	N/A	N/A	**83.46**
	F-measure	84.96	48.65	83.63	83.94	84.94	**85.28**	N/A	N/A	**89.29**
U2R	Recall	14.47	13.16	14.91	10.09	**15.79**	13.20	11.84	11.40	**72.8**
	Precision	9.35	2.05	5.47	53.49	**61.02**	**71.43**	55.10	53.06	7.98
	F-measure	11.36	3.54	8.00	16.97	**25.09**	**22.28**	19.49	18.77	14.38
R2L	Recall	1.45	**62.74**	6.90	3.55	11.01	8.40	8.33	13.05	**33.45**
	Precision	30.32	42.70	66.97	62.39	68.39	**98.84**	81.85	**82.37**	80.42
	F-measure	2.77	**50.82**	12.51	6.71	18.97	15.48	15.12	22.53	**47.24**
DOS	Recall	96.99	82.75	97.00	**97.56**	97.20	97.10	22.06	21.74	**98.83**
	Precision	99.69	94.00	99.42	99.86	**99.90**	**99.88**	95.75	96.68	99.8
	F-measure	98.32	88.02	98.19	**98.70**	98.53	98.47	35.86	35.50	**99.31**
PRB	Recall	81.88	**90.45**	81.61	86.27	88.60	83.30	81.16	**89.01**	86.25
	Precision	52.20	64.16	55.46	77.72	74.40	64.81	**77.92**	**82.11**	71.8
	F-measure	63.76	75.07	66.05	**81.77**	80.88	72.90	79.51	**85.42**	78.36

detection classification problem, therefore we will first review our past works on this field and afterwards present our main idea behind EFS-ACO.

We have previously worked on the well-known intrusion detection problem using the evolutionary fuzzy system approach [10, 34]. However, none of our past works were comparable to EFS-ACO. SRPP [10] was our first research activity in the field of utilizing evolutionary fuzzy systems to solve intrusion detection classification problem. In that paper we proposed a new fitness function for intrusion detection that its results were presented using a simplified and limited version of the train dataset which we have used in this paper.

The relation between [34] and EFS-ACO is that

Table 10. Comparison of Two Hybrid Evolutionary Fuzzy Systems for Intrusion Detection Classification Problem.

Algorithm	Time Complexity
EFS-ACO	$O(C \times R \times F \times S + A \times S)$
Hybrid-EFS	$O(C \times R \times F \times A \times S^2)$

C: Number of Classes in the classification problem

R: Number of Rules in the main population of the algorithm

F: Number of Features of the classification problem

S: Number of Samples of the dataset

A: Number of Ants in the local search algorithm

[34] holds an old idea which hybridizes ACO and GA by embedding a boosting local searcher to the main structure of the evolutionary process of GA. Although this type of hybridization seemed to be interesting, nevertheless it wasn't successful when we test the final classifier with the well known KDD-Cup 99 test data that has more than 300000 records. The new idea in EFS-ACO has solved this problem by linking the local searcher (instead of embedding it) to the structure of the evolutionary process of GA.

According to the above discussions, the main disadvantage of our previous works was that we have not emphasized on the main challenge of the intrusion detection problem which is finding accurate rules and making a suitable cooperation between them concurrently. In [10] we have only presented a new and albate simple one-phase algorithm named SRPP that was only capable of detecting simple intrusive behaviors. The new improved Hybrid-EFS [34] was a more accurate classification system which was tested on a 5 class intrusion detection problem (like EFS-ACO) but the used dataset for evaluating results was not the same as EFS-ACO since Hybrid-EFS was a time consuming approach and could not address complex intrusive behaviors. This disadvantage was because of its bad hybridization which was discussed earlier. In EFS-ACO we have linked the ACO local searcher to the main evolutionary fuzzy system (instead of embedding it like [34]). Using this kind of hybridization makes EFS-ACO much faster as mentioned in Table 10.

According to Table 10, we can see that the time complexity of EFS-ACO is $O(C \times R \times F \times S + A \times S)$ which means that since the hybridization of EFS-ACO links the ACO-based local searcher to the main evolutionary process of the learning system, its time complexity decreases significantly comparing with Hybrid-EFS in which the ACO-based local searcher is embedded in the main structure of the learning algorithm.

8 Conclusions

In this paper, a novel two-stage learning algorithm to extract fuzzy classification rules is introduced. The proposed learning algorithm hybridizes an evolutionary fuzzy system with an ant colony optimization procedure to find a suitable classification system for misuse intrusion detection. The capability of the resulting hybrid fuzzy system is investigated according to the well-known KDD-Cup99 benchmark dataset. Computer simulations on this dataset demonstrated that the added ACO-based local search stage would continue the learning process of the primary evolutionary fuzzy system efficiently. Moreover, the results indicate that in comparison to several traditional and new techniques, such as C4.5, Nave Bayes, k-NN and SVM and MOGF-IDS the proposed hybrid approach achieves higher classification accuracy and lower classification cost for the intrusion detection classification problem. Therefore, the resulted fuzzy classification rules can be used to produce a reliable intrusion detection system.

It would be interesting to investigate the hybridization of other meta-heuristics like artificial immune system and particle swarm optimization on the performance of evolutionary fuzzy systems. Moreover, the use of multi-objective evolutionary fuzzy systems to extract a comprehensible fuzzy classifier for classification problems is another interesting investigation topic, which is left for our future work.

Acknowledgements

The support of Iran Telecommunication Research Center for this work which is a result of a Ph.D. thesis in Sharif University of Technology is gratefully acknowledged. Moreover, the authors would like to thank the anonymous reviewers' useful comments. Finally we respect Mr. Emad Soroush's efforts at the first stages of this work.

References

[1] Giorgio Giacinto, Fabio Roli, and Luca Didaci. Fusion of Multiple Classifiers for Intrusion Detection in Computer Networks. *Pattern Recognition Letters*, 24(12):1795–1803, 2003. ISSN 0167-8655.

[2] Nong Ye, Qiang Chen, and C. M. Borror. EWMA Forecast of Normal System Activity for Computer Intrusion Detection. *IEEE Transactions on Reliability*, 53(4):557–566, 2004.

[3] Stefan Axelsson. Intrusion Detection Systems: A Survey and Taxonomy. Technical Report 99-15, Department of Computer Engineering, Chalmers

University of Technology, Sweden, March 2000.

[4] Norbik Bashah Idris and Bharanidlran Shanmugam. Artificial Intelligence Techniques Applied to Intrusion Detection. *Annual IEEE INDICON*, pages 52–55, 2005.

[5] Sung-Bae Cho. Incorporating Soft Computing Techniques into a Probabilistic Intrusion Detection System. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 32(2):154–160, 2002.

[6] Jun-feng Tian, Yue Fu, Ying Xu, and Jian-ling Wang. Intrusion Detection Combining Multiple Decision Trees by Fuzzy logic. In *Proceedings of the 6th International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT '05)*, pages 256–258, Washington, DC, USA, 2005. IEEE Computer Society.

[7] Sanghyun Cho and Sungdeok Cha. SAD: Web Session Anomaly Detection Based on Parameter Estimation. *Computers & Security*, 23(4):312–319, 2004.

[8] Hai-Hua Gao, Hui-Hua Yang, and Xing-Yu Wang. Ant Colony Optimization Based Network Intrusion Feature Selection and Detection. In *Proceedings of the 4th International Conference on Machine Learning and Cybernetics*, Guangzhou, China, 2005.

[9] Tansel Özyer, Reda Alhajj, and Ken Barker. Intrusion Detection by Integrating Boosting Genetic Fuzzy Classifier and Data Mining Criteria for Rule Pre-Screening. *Journal of Network and Computer Applications*, 30(1):99–113, 2007.

[10] Mohmmad Saniee Abadeh, Jafar Habibi, and Caro Lucas. Intrusion Detection Using a Fuzzy Genetics-Based Learning Algorithm. *Journal of Network and Computer Applications*, 30(1):414–428, 2007.

[11] Stefan Axelsson. The Base-Rate Fallacy and the Difficulty of Intrusion Detection. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):186–205, 2000.

[12] Suseela T. Sarasamma, Qiuming A. Zhu, and Julie Huff. Hierarchical Kohonenen Net for Anomaly Detection in Network Security. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 35(2):302–312, 2005.

[13] Yong Feng, Zhong-Fu Wu, Kai-Gui Wu, Zhong-Yang Xiong, and Ying Zhou. An Unsupervised Anomaly Intrusion Detection Algorithm Based on Swarm Intelligence. In *Proceedings of the 4th International Conference on Machine Learning and Cybernetics*, Guangzhou, China, 2005.

[14] Ahmed Awad E. Ahmed and Issa Traore. Anomaly Intrusion Detection Based on Biometrics. In *Proceedings of the 2005 IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, USA.

[15] KDD-Cup Data Set. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[16] Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. Mining Audit Data to Build Intrusion Detection Models. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 66–72. AAAI Press, 1998.

[17] James Cannady. Artificial Neural Networks for Misuse Detection. In *National Information Systems Security Conference*, pages 368–81, 1998.

[18] Hervé Debar and Bernadette Dorizzi. An Application of a Recurrent Network to an Intrusion Detection System. In *International Joint Conference on Neural Networks (IJCNN)*, volume 2, pages 478–483, Baltimore, MD, USA, 1992.

[19] Hervé Debar, Monique Becker, and Didier Siboni. A Neural Network Component for an Intrusion Detection System. In *Proceedings of the 2nd IEEE Symposium on Security and Privacy (SP '92)*, pages 240–250, Washington, DC, USA, 1992. IEEE Computer Society.

[20] Srinivas Mukkamala and Andrew H. Sung. Feature Selection for Intrusion Detection Using Neural Networks and Support Vector Machines. *Journal of the Transport Research Board*, (1822):33–39, 2003.

[21] Martin Riedmiller and Heinrich Braun. A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In *Proceedings of the 2nd IEEE International Conference on Neural Networks*, pages 586–591, San Francisco, USA, 1993.

[22] Amit Kumar Choudhary and Akhilesh Swarup. Neural Network Approach for Intrusion Detection. In *Proceedings of the 2nd ACM International Conference on Interaction Sciences (ICIS '09)*, pages 1297–1301, Seoul, South Korea, 2009.

[23] Jake Ryan, Meng jang Lin, and Risto Miikkulainen. Intrusion Detection with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 10, pages 943–949. MIT Press, 1998.

[24] Susan M. Bridges and Rayford B. Vaughn. Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection. In *Proceedings of the National Information Systems Security Conference (NISSC)*, pages 13–31, 2000.

[25] Steven A. Hofmeyr, Stephanie Forrest, and Anil Somayaji. Intrusion Detection using Sequences of System Calls. *Journal of Computer Security*, 6:151–180, 1998.

[26] Dipankar Dasgupta and Fabio González. An Immunity-Based Technique to Characterize Intrusions in Computer Networks. *IEEE Transactions on Evolutionary Computation*, 6(3):1081–

1088, 2002.

[27] Paul K. Harmer, Paul D. Williams, Gregg H. Gunsch, and Gary B. Lamont. An Artificial Immune System Architecture for Computer Security Applications. *IEEE Transactions on Evolutionary Computation*, 6(3):252–280, 2002.

[28] Xiang-Rong Yang, Jun-Yi Shen, and Rui Wang. Artificial Immune Theory Based Network Intrusion Detection System and the Algorithms Design. In *Proceedings of the 1st IEEE International Conference on Machine Learning and Cybernetics*, volume 1, pages 73–77, 2002.

[29] Chi-Ho Tsang, Sam Kwong, and Hanli Wang. Anomaly Intrusion Detection Using Multi-Objective Genetic Fuzzy System and Agent-Based Evolutionary Computation Framework. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM '05)*, pages 789–792, Washington, DC, USA, 2005. IEEE Computer Society.

[30] Baoguo Xu and Apin Zhang. Application of support Vector Clustering Algorithm to Network Intrusion Detection. In *International Conference on Neural Networks and Brain (ICNN&B '05)*, volume 2, pages 1036–1040, 2005.

[31] Sang Hyun Oh and Won Suk Lee. An Anomaly Intrusion Detection Method by Clustering Normal User Behavior. *Computers & Security*, 22(7): 596–612, 2003.

[32] Elizabeth Leon, Olfa Nasraoui, and Jonatan Gomez. Anomaly Detection based on Unsupervised Niche Clustering with Application to Network Intrusion Detection. In *Proceedings of the 6th IEEE Congress on Evolutionary Computation (CEC2004)*, volume 1, pages 502–508, 2004.

[33] Yu Guan, Ali A. Ghorbani, and Nabil Belacel. Y-Means: A Clustering Method for Intrusion Detection. In *Canadian Conference on Electrical and Computer Engineering*, pages 1083–1086, 2003.

[34] Mohammad Saniee Abadeh, Jafar Habibi, and Emad Soroush. Induction of Fuzzy Classification Systems via Evolutionary ACO-Based Algorithms. *International Journal of Simulation Systems, Science & Technology*, 9(3):1–8, 2008.

[35] Hisao Ishibuchi, Ken Nozaki, and Hideo Tanaka. Distributed Representation of Fuzzy Rules and its Application to Pattern Classification. *Fuzzy Sets and Systems*, 52(1):21–32, 1992.

[36] Tomoharu Nakashima Hisao Ishibuchi. Improving the Performance of Fuzzy Classifier Systems for Pattern Classification Problems with Continuous Attributes. *IEEE Transactions on Industrial Electronics*, 46(6), 1999.

[37] Marco Dorigo. *Optimization, Learning, and Natural Algorithms (In Italian)*. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1992.

[38] Marco Dorigo and Luca Maria Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, 1(1): 53–66, 1997.

[39] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26:29–41, 1996.

[40] D. Costa and A. Hertz. Ants Can Colour Graphs. *Journal of the Operational Research Society*, 48 (3):295–305, 1997.

[41] Luca Maria Gambardella and Marco Dorigo. Ant Colonies for the Quadratic Assignment Problem. *Journal of the Operational Research Society*, 50 (2):167–176, 1999.

[42] Vittorio Maniezzo and Alberto Colorni. The Ant System Applied to the Quadratic Assignment Problem. *IEEE Transactions on Knowledge and Data Engineering*, 11(5):769–778, 1999.

[43] Bernd Bullnheimer, Richard F. Hartl, and Christine Strauss. An Improved Ant System Algorithm for the Vehicle Routing Problem. *Annals of Operations Research*, 89:319–328, 1999.

[44] Luca Maria Gambardella, Éric Taillard, and Giovanni Agazzi. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. *New Ideas in Optimization*, 52(2):63–76, 1999.

[45] ChristineSolnon. Ants Can Solve Constraint Satisfaction Problems. *IEEE Transactions on Evolutionary Computation*, 6(4):347–357, 2002.

[46] Rafael S. Parpinelli, Heitor S. Lopes, and Alex A. Freitas. Data Mining with an Ant Colony Optimization Algorithm. *IEEE Transactions on Evolutionary Computation*, 6(4):321–332, 2002.

[47] Lincoln Laboratory MIT. http://www.ll.mit.edu.

[48] Charles Elkan. Results of the KDD'99 Classifier Learning. *ACM SIGKDD Explorations Newsletter*, 1(2):63–64, 2000.

[49] Ramesh Agarwal and Mahesh V. Joshi. PNrule: A New Framework for Learning Classifier Models in Data Mining (A Case-Study in Network Intrusion Detection). In *Proceedings of the 1st SIAM Conference on Data Mining*, Chicago, IL, USA, 2001.

Mohammad Saniee Abadeh received his B.S. degree in Computer Engineering from Isfahan University of Technology, Isfahan, Iran, in 2001, the M.S. degree in Artificial Intelligence from Iran University of Science and Technology, Tehran, Iran, in 2003 and his Ph.D. degree in Artificial Intelligence at the Department of Computer Engineering in Sharif University of Technology, Tehran, Iran in February 2008. Dr. Saniee Abadeh is currently a faculty member at the Faculty of Electrical and Computer Engineering at Tarbiat Modares University. His research has focused on developing advanced meta-heuristic algorithms for data mining and knowledge discovery purposes. His interests include computational intelligence, evolutionary algorithms, fuzzy genetic systems, memetic algorithms, biological computing, and data mining.

Jafar Habibi received his B.S. degree in computer engineering from the Supreme School of Computer, his M. S. degree in industrial engineering from Tarbiat Modares University and his Ph.D. degree in Computer engineering from Manchester University. At present, he is a faculty member at the computer engineering department at Sharif University of Technology and the managing directorputing Machine Services. He is supervisor of Sharif's Robo-Cup Simulation Group. His research interests are mainly in the areas of computer engineering, simulation systems, MIS, DSS and evaluation of computer systems performance.